grammar.y

```
%{
#include <stdio.h>
%}

%token START
%token FINISH
%token DEF
%token NUMBER
%token STRING
%token CHAR
%token ARRAY
%token OF
%token UNDEFINED
%token READ
%token IF
%token STARTIF
%token FINISHIF
%token ASSIGN
%token WHILE
%token STARTWHILE
%token FINISHWHILE
%token PROC
%token STARTPROC
%token FINISHPROC
%token CALL
%token RETURN
%token PRINT
%token id
%token constant
%token Epsilon
```

```
%token NEGPOSDIGIT
%token ERRORNUMCONST
%token OPERATOR
%token SEPARATOR
%token SPACE

%%
program: START cmds FINISH
                ;
cmds: cmd cmdsconf
                ;
cmdsconf: Epsilon
                | cmds
                ;
cmd: simplecmd
                | structcmd
                ;
simplecmd: defcmd
                | assigncmd
                | readcmd
                | printcmd
                | returncmd
                ;
defcmd: DEF declist
                ;
declist: declaration declistconf
                ;
declistconf: Epsilon
                | ';' declist
                ;
```

```
declaration: id ':' dtype
                ;
dtype: primitive
                | arraydecl
                ;
primitive: NUMBER
                | STRING
                | CHAR
                ;
arraydecl: ARRAY '[' arraydeclconf
                ;
arraydeclconf: constant ']' OF primitive
                | id ']' OF primitive
                ;
assigncmd: ASSIGN id ':' assigncmdconf
                ;
assigncmdconf: symbolvalue
                | '(' expressionstart ')'
                | UNDEFINED
                ;
symbolvalue: id symbolvalueid
                | constant
                | '[' symbolvalueconf
                ;
symbolvalueid: Epsilon
                | '[' symbolvalueconf
                ;
symbolvalueconf: id ']'
                | constant ']'
                ;
expressionstart: term expression
```

```
                ;
expression: '+' term expression
                | '-' term expression
                | Epsilon
                ;
term: factor muldiv
                ;
muldiv: '*' factor muldiv
                | '/' factor muldiv
                | Epsilon
                ;
factor: '(' expressionstart ')'
                | symbolvalue
                ;
readcmd: READ id readcmdconf
                ;
readcmdconf: Epsilon
                | '[' symbolvalueconf
                ;
printcmd: PRINT '(' expressionprint ')'
                ;
expressionprint: factorprint expressionprintconf
                ;
expressionprintconf: Epsilon
                | '+' expressionprint
                ;
factorprint: id
                | constant
                | callstmt
                ;
returncmd: RETURN returncmdconf
```

```
                ;
returncmdconf: expressionstart
                | callstmt
                ;
structcmd: ifstmt
                | whilestmt
                | procstmt
                | callstmt
                ;
ifstmt: IF condition STARTIF cmds FINISHIF
                ;
condition: basiccondition conditionconf
                ;
conditionconf: Epsilon
                | logicaloperator condition
                ;
basiccondition: symbolvalue comparisonoperator basicconditionconf
                ;
basicconditionconf: symbolvalue
                | UNDEFINED
                ;
comparisonoperator: "<"
                | ">"
                | "<="
                | ">="
                | "==="
                | "!=="
                ;
logicaloperator: "&&"
                | "||"
                ;
```

```
whilestmt: WHILE condition STARTWHILE cmds FINISHWHILE
                ;
procstmt: PROC id '(' procstmtconf
                ;
procstmtconf: ')' STARTPROC cmds FINISHPROC
                | declist ')' STARTPROC cmds FINISHPROC
                ;
callstmt: CALL id '(' paramslist ')'
                ;
paramslist: expressionstart paramslistconf
                | Epsilon
                ;
paramslistconf: Epsilon
                | ',' paramslist
                ;
%%
int main(int argc, char **argv)
{
  yyparse();
}

int yyerror(char *s)
{
  fprintf(stderr, "error: %s\n", s);
}
```